UNITED STATES PATENT APPLICATION

FOR

METHOD AND APPARATUS FOR ENTITY REMOVAL FROM A CONTENT MANAGEMENT SOLUTION IMPLEMENTING TIME-BASED FLAGGING FOR CERTAINTY IN A RELATIONAL DATABASE ENVIRONMENT

BY

DONALD E. BENSON,

EDWARD J. GALLAGHER,

MANG-RONG HO,

AND

DWAYNE L. RICHARDSON

DESCRIPTION

Field

[001] The present invention relates to managing content for an enterprise. In particular, the present invention relates to managing removal of content for an enterprise.

Background

[002] Today, many enterprises store a variety of types of content for its operations. Electronically implemented "libraries" have become a popular way for an enterprise to manage its content. It is common for these content management solutions to maintain and control thousands of different entities, such as files, objects, images, or documents. In addition, these content management solutions may maintain and control multiple versions of this content.

[003] One problem with allowing multiple versions of content is controlling how and when a user updates or deletes one or more versions of that content. For example, a user may prematurely update or delete a valid version of an image stored in a content management system. As another example, a user may inadvertently update an old or deleted version of a document rather than the most current version of that document. These actions may thus cause a content management system to have one or more invalid versions of an entity, or make the content management system unable to locate a proper version of an entity.

[004] Accordingly, it may be desirable to provide methods and apparatus that manage multiple versions of content.

SUMMARY

[005] In accordance with one feature of the invention, a request that affects an item is validated. A version of the item is identified based on a first time. Whether the request affects an object associated with the item is determined. When the request affects an object, a version of the object is identified based on a second time. The request is then completed based on the version of the item and the version of the object.

[006] In accordance with another feature of the invention, requests that delete an item are processed, wherein the items may include associated objects. When a request that deletes an item is received, a first age is identified for the item affected by the request. Information indicating a second age of at least one associated object is then retrieved. The item and the at least one associated object are deleted based on whether the first age of the item is greater than or equal to the second age of the at least one associated object

[007] Additional features of the invention will be set forth in part in the description which follows, and in part will be obvious from the description, or may be learned by practice of the invention. The features of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims.

[008] It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention, as claimed.

BRIEF DESCRIPTION OF THE DRAWINGS

[009] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate embodiments of the invention and together with the description, serve to explain the principles of the invention.

[010] Figure 1 shows a content management system that is consistent with the principles of the present invention;

[011] Figure 2 shows a conceptual diagram of a library server that is consistent with the principles is invention;

[012] Figure 2A shows one example of tables that may be used by the library server in accordance with the principles of the present invention;

[013] Figure 3 shows a conceptual diagram of a resource manager that is consistent with the principles of the present invention;

[014] Figure 3A shows one example of a table that may be used by the resource manager in accordance with the principles of the present invention; and

[015] Figure 4 shows a flow diagram for managing multiple versions of content in accordance with the principles of the present invention.

DESCRIPTION OF THE EMBODIMENTS

[016] One feature of the present invention provides a content management system that can handle multiple versions of content. Reference will now be made in detail to exemplary embodiments of the invention, which are illustrated in the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

[017] Figure 1 shows a content management system **100** that is consistent with the principles of the present invention. As shown, content management system **100** may comprise a client **102**, a library server **104**, and a resource manager **106**. These components may be coupled together using one or more networks, such as a local area network, or wide area network. In addition, these components may communicate with each other using known protocols, such as the transport control protocol and internet protocol ("TCP/IP") and hypertext transport protocol ("HTTP").

[018] The components of content management system **100** may be implemented on separate devices or may be implemented on one or more of the same devices or systems. For example, library server **102** and resource manager **104** may be installed on the same machine and run under a common operating system. Alternatively, content management system **100** may have one or more of its components implemented on multiple machines that run different operating systems. Some of the specific components of content management system **100** will now be described.

[019] Client **102** provides a user interface for content management system **100**. Client **102** may be implemented using a variety of devices and software. For example client **102** may be implemented on a personal computer, workstation, or terminal. In addition, client **102** may run under a Windows® operating system, or through a browser application, such as Internet Explorer™ by Microsoft® Corporation or Netscape Navigator™ by Netscape Communications® Corporation. Although Figure 1 shows a single client, content management system **100** may include any number of clients.

[020] Library server **104** stores, manages, and provides access control to items stored by content management system **100**. Library server **104** processes requests, such as creates, reads, updates, and deletes, from client **102** and maintains the data integrity between the other components of content management system **100**, such as resource manager **106**. For example, library server **104** may work in conjunction with resource manager **106** to retrieve an object, such as a document or image file, that is referenced by an item.

[021] Library server **104** may be implemented using a variety of devices and software. For example, library server **104** may be a computer that runs one or more application programs and stored procedures under an operating system, such as z/OS®, Windows®, AIX®, or Solaris®. In addition, library server **104** may include a database management system, such as a relational database management system, to manage stored items and perform searches for content management system **100**. For example, library server **104** may use the DB2® Universal Database™ by International Business Machines Corporation (IBM®). Library server **104** is also described with reference to Figure 2.

[022] Resource manager **106** stores objects corresponding to items in content management system **100**. Objects may be any data entity for an item that is in digital form. For example, an object may be an audio file, an application, an image, text, or a video file. Resource manager **106** may store the objects in various formats, such as JPEG images, MP3 audio, AVI video, and ASCII text. Resource manager **106** may also store objects in formats, such as Microsoft® Word, Lotus® Word Pro®, and Wordperfect®.

[023] Furthermore, resource manager **106** may also be configured to store multiple copies of objects on the same or a separate resource manager (not shown). Although Figure 1 shows a single resource manager, content management system **100** may include any number of resource managers. For example, content management system **100** may include multiple resource managers that are distributed across one or networks.

[024] Resource manager **106** may be implemented using known devices and software. For example, resource manager **106** may be installed on one or more computers that run under the z/OS® operating system, and includes a DB2® Universal Database™, as well as a server to communicate with client **102** and library server **104**, such as a HTTP server. In addition, resource manager **106** may include one or more storage devices, such as a magnetic disc drive. Resource manager **106** is also described with reference to Figure 3.

[025] Fig 2 shows a conceptual diagram of library server **104** that is consistent with the principles of the present invention. As shown, library server **104** may comprise an application program **200** and a library server database **202**.

[026] Application program **200** is program code that implements the functions and procedures and library server **104**, such as communications with client **102** and resource manager **106** and operations with library server database **202**. Application program **200** may be written in a variety of host programming languages, such as C, C++, Java, or COBOL. In addition, application program **200** may include a set of embedded

modules **206** that comprise SQL statements for interacting with library server database **202**.

[027] Library server database **202** serves as a catalog for items stored by content management system **100**. In order to catalog a variety items, library server database **202** may classify items according to an item identifier. Library server **202** may automatically assign the item identifier when the item is stored or updated. In addition, in order to allow for multiple versions of an item, library server **202** may classify items according to a version and timestamp.

[028] Objects associated with a particular item, such as a document, may also be indexed by library server database **202** and stored by resource manager **106**. For example, for an insurance business, library server database **202** may use an item type for insurance claims and policy holders. The item type specifies the format of the information, such as the policy holder name, address, and vehicle information. Each individual claim and policy holder would then be considered an item and indexed by library server database **202**. Documents corresponding to each individual claim, such as a fax, may then be stored as objects in resource manager **106**. Objects may also have multiple versions.

[029] Library server database **202** may be implemented using a variety of devices and software. For example, library server database **202** may be implemented as a relational database, such as a DB2® Universal Database™. In addition, library server database **202** may use a variety of types of storage, such as tape drive, optical storage units, or magnetic disk drive.

[030] Library server database **202** may use a set of tables, such as an index table **204** and transaction table **206**. Index table **204** may contain information that indexes the items stored by content management system **100**. For example, index table **204** may index or reference objects stored by resource manager **106** for a particular item. Transaction table **206** may contain information for controlling of transactions to be performed by content management system **100**. Transaction table **206** may be useful to ensure that transactions that have been concurrently requested to library server database **202** are properly controlled. For example, transaction table **206** may contain information tracking items that have been deleted (or updated) by one or more users. One example of index table **204** and transaction table **206** is further described with reference to Figure 2A.

[031] Referring now to Figure 2A, examples of index table **204** and transaction table **206** are shown. As noted, index table **204** provides information that indexes the items stored by content management system **100**. For example, index table **204** allows library server **104** to locate one or more objects stored in resource manager **106**, which correspond to a particular item. As shown, index table **204** may comprise an item identifier column **208**, a timestamp column **210**, and one or more value columns 226.

[032] Item identifier column **208** includes information that uniquely identifies each item. An item identifier may be a numeric or alphanumeric sequence that is automatically assigned by library server database **202**.

[033] Timestamp column **210** includes information that specifies a time for a particular item. The timestamps may be in a variety of formats and use numeric or

alphanumeric characters. Library server **104** may assign the timestamps automatically for content management system **100** to ensure that a single time reference is used for all transactions. Library server **104** may assign the timestamps according to any level of accuracy, such as to each millisecond.

[034] Value columns **212** include information that indicates various attributes of an item. Value columns **212** may include information in any format, such as numeric, and alphanumeric characters. The information in value columns **212** may, for example, include information that describes certain characteristics or properties of an item, such as a first name, surname, age, or city. The information in value columns **212** may also be used as key fields. For example, information in value columns **212** may be used to reference or location objects that are stored in resource manager **106**.

[035] In addition, value columns **212** may include information that indicates a version for a particular item. A version may be identified by a numeric or alphanumeric sequence. This version sequence may be automatically assigned by library server database **202** or set by a user.

[036] Transaction table **206** includes information for controlling transactions performed by content management system **100**. A transaction may affect only the information in library server database **202**, or information in both library server database **202** and content database **302**. For example, a transaction may metadata associated with an item and corresponding content, such as objects, documents, or images, of an item. As shown, transaction table **206** may comprise an item identifier column **214**, a timestamp column **216**, and one or more transaction attribute columns **218**.

[037] Item identifier column **214** includes information that uniquely identifies each item affected by a particular transaction. As shown, the information in item identifier column **214** may serve as a key to link information in transaction table **206** with information in index table **204**. For example, item identifier column **214** may use the same identifiers contained in item identifier column **208** of index table **204**. The item identifiers in item identifier column **214** may be in a variety of formats, such as numeric or alpha numeric. Alternatively, the item identifiers in column **214** may be assigned automatically by library server database **202** based on the identifiers used in item identifier column **208** of index table **204**.

[038] Timestamp column **216** indicates the timestamp of the particular item that is affected by a transaction. For example, when a delete transaction is requested, library server database **104** may write the item's timestamp from column **210** of index table **204** into timestamp column **216**.

[039] Transaction attribute columns **218** provide one or more columns of information indicating the attributes and parameters of a particular transaction. For example attribute columns **218** may include information about the types of access controls enforced for each item stored by content management system **100**, isolation levels, references, unique attributes, input parameters, and output parameters of a particular transaction. In addition, transaction attribute columns **218** may include the version identifier of the item affected by the transaction.

[040] Figure 3 shows a conceptual diagram of resource manager **106** that is consistent with the principles of the present invention. As shown, resource manager **106**

may comprise a communication server **300**, a content database **302**, and a recovery module **304**.

[041] Server **300** provides communication services between resource manager **106**, client **102** and library server **104**. In one embodiment, communication server **300** is implemented as an HTTP server that is configured to communicate with client **102** and library server **104**.

[042] Content database **302** manages and stores objects for content management system **100**. Content database **302** may be implemented using a variety devices and software. For example, in one embodiment content database **302** implemented as a relational database, such as DB2® Universal Database™. In addition, content database **302** may use a variety of types of storage, such as can drive optical storage units, or magnetic disk drive.

[043] In addition, content database **302** may include one or more tables, such as a content index table **306**. Content index table **306** may contain information that indexes the content corresponding to various items. For example, content index table **306** may index or reference objects, such as documents and image files, stored by resource manager **106**. An example of content index table **306** is further described with reference to Figure 3A.

[044] Recovery module **304** is program code that allows for recovery of transaction errors. For example, recovery module **304** may assist in recovering data that was inadvertently deleted. In addition, when content management system **100** includes multiple resource managers, recovery module **304** may be used to coordinate data

recovery in the event of a transaction error. Recovery module **304** may be written in a variety of host programming languages, such as C, C++, Java, or COBOL. Recovery module **304** may be a separate application within resource manager **106**, or may be embedded as a component of communications server **300**.

[045] Referring now to Figure 3A, an example of content index table **306** is shown. As shown, content index table **204** may comprise an object identifier column **308**, an item identifier column **310**, a timestamp column **312**, and one or more value columns **314**.

[046] Object identifier column **308** includes information that uniquely identifies each object. An object identifier may be a numeric or alphanumeric sequence that is automatically assigned by library server database **202**.

[047] Item identifier column **310** includes information that uniquely identifies each item to which a particular object corresponds. An item identifier may be a numeric or alphanumeric sequence that is automatically assigned by library server database **202**. The information in column **310** may correspond to the item identifiers used in index table **204** and transaction table **206**, respectively. Accordingly, item identifier column **310** may serve as a key that links the items indexed by library server database **202** with their corresponding objects stored in content database **302**.

[048] Timestamp column **312** includes information that specifies a time for a particular object stored in content database **302**. The timestamps may be in a variety of formats and use numeric or alphanumeric characters. In one embodiment, library server **104** may assign the timestamps for each object automatically to ensure that a single time

reference is used for objects. Library server **104** may assign the timestamps for each object according to any level of accuracy, such as to each millisecond.

[049] Value columns **314** include information that indicates various attributes of each object. Value columns **314** may include information in any format, such as numeric, and alphanumeric characters. The information in value columns **314** may, for example, include information that describes certain characteristics or properties of an object, such as whether the object is text searchable, its location, and data format.

[050] In addition, value columns **314** may include information that indicates a version for a particular object. A version may be identified by a numeric or alphanumeric sequence. This version sequence may be automatically assigned by library server database **202**, content database **302**, or set by a user.

[051] Figure 4 shows a flow diagram for dynamically constructing SQL statements in accordance with the principles of the present invention. In stage **400**, content management system **100** receives a request from a user. For example, a user may operate client **102** to create, read, update, or delete an item or item type, using a browser application or filling out an online form. Client **102** may then gather this information and forward the request to library server **104**.

[052] In stage **402**, content management system **100** routes the request to library server **104** and initiates processing of the request. Library server **104** may then parse the request's contents to identify the item affected by the request. For example, library server **104** may run application program **200** to identify the item affected by the

request. An item may be identified by its timestamp in timestamp column **210**, its version as indicated by information in value columns **212**, or a combination of both.

[053] Application program **200** may then use one or more SQL statements to initiate processing for the request. For example, application program **200** may write an entry into transaction table **206**. In particular, application program **200** may write the item's identifier into column **214**, the timestamp for that item into timestamp column **216**, and any appropriate attribute information into transaction attribute columns **206**, such as the item's version information.

[054] Transaction table **206** may be tailored for specific types of requests. For example, transaction table **206** may be used to track items in which a user has requested a delete transaction. As another example, transaction table **206** may be used to track new items created by the user. Any number of transaction tables may be used in library server **104**.

[055] In stage **404**, application program **200** determines whether the transaction affects only library server database **202** or both library server database **202** and content database **302**. For example, if the request includes an update to an item's content or the request deletes an item that includes objects stored in content database **302**, then application program **200** may decide that the request affects both the library server database **202** and content database **302**. If the request affects information only in library server **202**, then processing flows to stage **406**. If the request affects information in both library server database and content database **302**, then processing flows to stage **408**.

[056] In stage **406**, the request only affects information in library server database **202**. Accordingly, application program **200** operates in conjunction with library server **202** to process the request. Upon completing the request, library server **202** may then notify application program **200**. Application program **200** may then notify the user at client **102** that the request has been completed.

[057] In stage **408**, the request affects information in both library server database **202** and content database **302**. Accordingly, application program **200** and/or library server database **202** may forward information to resource manager **106**. For example, library server database **202** may forward information from transaction table **206**, such as timestamp information from column **216** and transaction attributes from column **218**. Resource manager **106** may then receive this information through communications server **300** and forward it to content database **302**.

[058] In stage **410**, content database **302** determines whether the request is valid. In particular, content database **302** may analyze the transaction's attributes and identify one or more objects affected by the request. For example, content database **302** may query content index table **306** and retrieve objects that are associated with a particular item's identifier. In addition, content database **302** may retrieve an object's timestamp and version identifier from timestamp column **312** and value columns **314**, respectively.

[059] Content database **302** then compares the object's timestamp to the timestamp provided from library server database **202**, i.e., from timestamp column **216** of transaction table **206**. Content database **302** may also compare an item's version

identifier with each object's version identifier. Content database **302** may then decide whether the request is valid based on the timestamps. For example, content database **302** may decide that a request is valid based on checking the timestamps. Alternatively, content database **302** may decide that a request is valid based on checking both the timestamps and version identifiers.

[060] A request may be considered valid under a variety of circumstances. For example, if the timestamp of an item matches the timestamp of the object, then content database **302** may consider the request valid. As another example, if both the timestamps and version identifiers of the item and its objects match, then content database **302** may consider the request valid. As yet another example, if the timestamp of the item is less than the timestamp of the object, then this may indicate that the item has been updated, such as when the user has uploaded a new document. However, if the user has requested that the item be deleted, then content database **302** may still consider this request valid, because the item is now obsolete and a new version of the item has likely been created in library server database **202**.

[061] Likewise, a request may be considered invalid under a variety of circumstances. For example, if the timestamp of the item is greater than the timestamp of an object, then this may indicate the object is obsolete. Also, if the timestamps match, but the version identifiers of the items and object do not match, then content database **302** may consider the request invalid.

[062] If the request is considered valid, the processing may flow to stage **412**. If the request is considered invalid, then processing may flow to stage **414**.

[063] In stage **412**, the request is valid, and thus, library database **202** and content database **302** may carry out their respective actions to process the request. For example, for a delete request, content database **302** may remove an object from its storage and update content index table **306**. When deleting an object, content database **302** may remove its entry from table **306**. Upon completing its actions, content database **302** may then send a message through communications server **300** back to library server database **202**. In response, library server database **202** may update transaction table **206** to indicate that the transaction has been completed. Alternatively, library server database **202** may write another entry into transaction table **206** to record when the transaction was completed.

[064] In addition, library server database **202** may update the timestamp recorded in timestamp column **210** of index table **204** to record when an item was last updated. Library server database **202** may also update the version identifier recorded in value columns **212** after a transaction is completed. Upon completing the request, library server database **202** may notify application program **200**. Application program **200** may then notify the user at client **102** that the request was completed.

[065] In stage **414**, the request is found invalid, and thus, content database **302** may generate one or more messages. For example, content database **302** may generate an error message to recovery module **304**. Recovery module **304** may then notify library server database **202** through communications server **300** of the error. Alternatively, content database **302** may notify library server **202** of an error directly.

[066] The invalid request may then be rolled back. In particular, upon receiving notice of an invalid request, library server database **202** may update transaction table **206**. For example, library server database **202** may remove the entry for the invalid transaction from transaction table **206**. In addition, library server database **202** may roll back any changes made to information in index table **204**. Library server database **202** may then notify application program of the invalid request. Subsequently, application program **200** may also notify the user at client **102** that the request was invalid and could not be performed.

[067] Other embodiments of the invention will be apparent to those skilled in the art from consideration of the specification and practice of the invention disclosed herein. It is intended that the specification and examples be considered as exemplary only, with a true scope and spirit of the invention being indicated by the following claims.